dV/dt: Accelerating the Rate of Progress towards Extreme Scale Collaborative Science

Miron Livny (UW)
Bill Allcock (ANL)
Ewa Deelman (USC)
Douglas Thain (ND)
Frank Wuerthwein (UCSD)

https://sites.google.com/site/acceleratingexascale

USC Viterbi School of Engineering
Information Sciences Institute

Funded by the Department of Energy

Goal: "make it easier for scientists to execute large-scale computational tasks that use the power of computing resources they do not own to process data they did not collect with applications they did not develop"

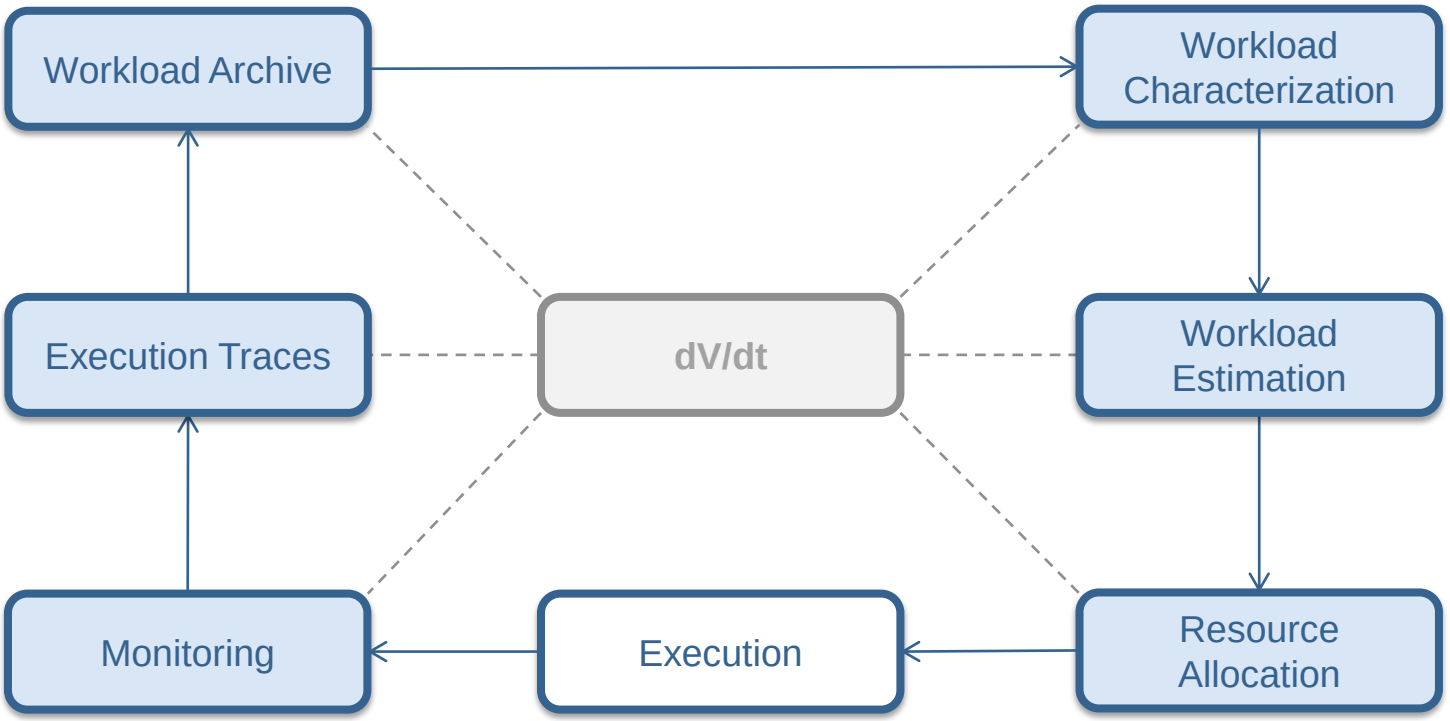
Challenges

- Estimate the application resource needs
- Allocate the needed resources
- Manage applications and resources during run, adapt allocations, or intervene on behalf of the resources

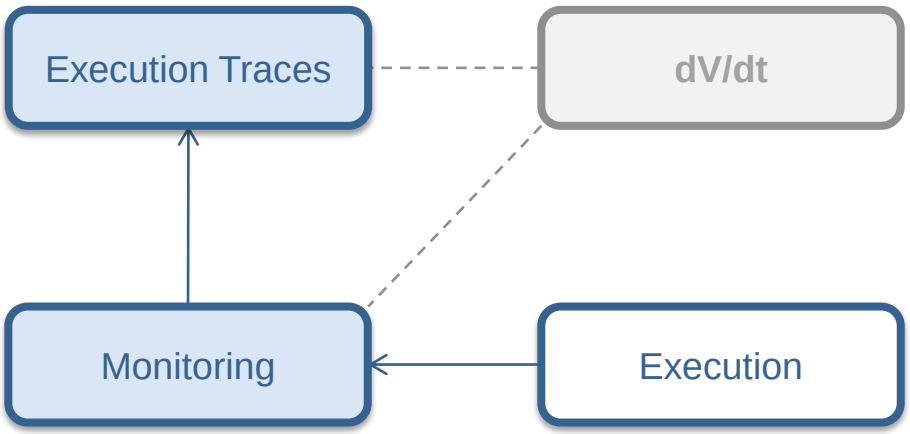
Experimental Foundation

- Real-world applications
  - Sets of tasks and workflows managed by workflow management systems (Pegasus and Makeflow)
- State of the art computing capabilities—Argonne Leadership Computing Facility and Open Science Grid
- Campus resources at ND, UCSD and UW
- Commercial cloud services
- Experimentation from the point of view of a scientist: "submit locally and compute globally"
- Pay attention to the cost involved in acquiring the resources and the human effort involved in software and data deployment and application management
  - Automate as much as possible

Approach


Workload Archive
Workload Characterization
Execution Traces
dV/dt
Workload Estimation
Monitoring
Execution
Resource Allocation

Monitoring Resource Usage



HTC Monitoring

- Job wrappers that collect information about processes
  - Runtime, peak disk usage, peak memory usage, CPU usage, etc.
- Mechanisms
  - Polling (not accurate, low overhead)
  - ptrace() system call interposition (accurate, high overhead)
  - LD_PRELOAD library call interposition (accurate, low overhead)
- Kickstart (Pegasus) and resource-monitor (Makeflow)

Error (Accuracy)

	Polling	LD_PRELOAD	Ptrace (syscalls)
CPU	0.5% - 12%	0.5% - 5%	< 0.2%
Memory	2% - 14%	< 0.1%	~ 0%
I/O	2% - 20%	0%	0%

Overhead

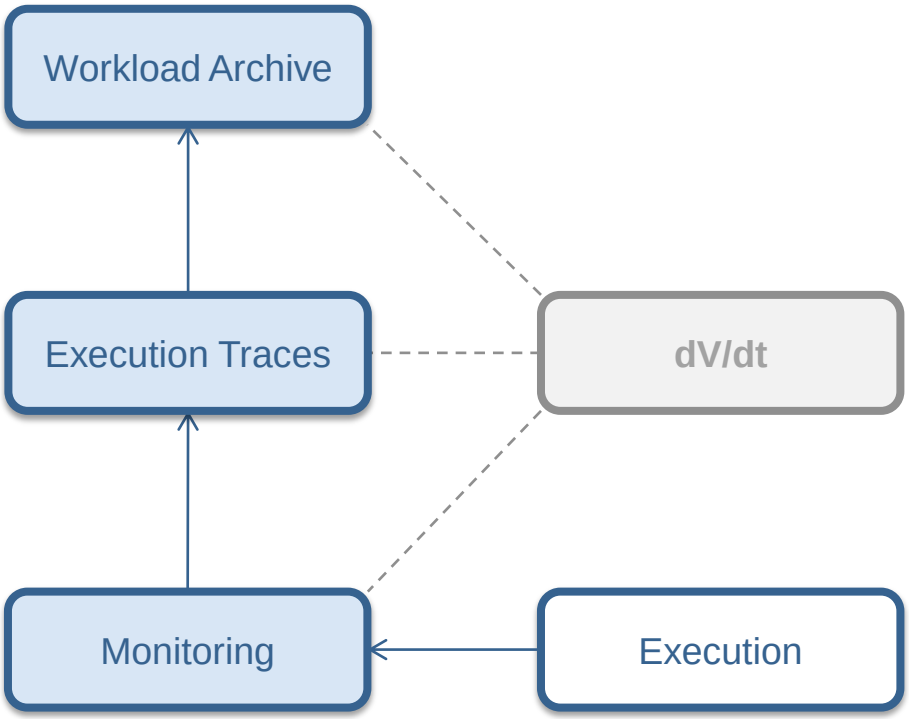
	Polling	LD_PRELOAD	Ptrace (syscalls)
CPU	low	low	low
Memory	low	medium	medium
I/O	low	low	high

Gideon Juve, et al., Practical Resource Monitoring for Robust High Throughput Computing, University of Southern California, Technical Report 14-950, 2014.

HPC Monitoring (ALCF)

- Job information from scheduler (Cobalt)
  - Use scheduler data for both scheduler and individual task data
  - Job runtime, number of cores, user estimates, etc.
- I/O using Darshan
  - Instrumentation automatically linked into codes at compile time
  - Captures POSIX I/O, MPI I/O and some HDF5 and NetCDF functions
  - Amount read/written, time in I/O, files accessed, etc.
  - Very low overhead in both time and memory
- Performance Counters using AutoPerf
  - Using built-in hardware performance counters
  - Also enabled at compile time
  - Counters zeroed in MPI_Init, and reported in MPI_Finalize
  - FLOPs, cache misses, etc.
  - Users can take control of performance counters preventing this from working

Building resource archives



Resources Archive

- The resource summary archive captures the information gathered by our monitoring tools
- The archive is publicly readable at http://dvdt.crc.nd.edu.
  - Build on top of the content management system Drupal and custom PHP and python code
  - Database backend running MySQL.
- Users of the archive can submit sets of resources summaries through a web interface, or with a batch job using ssh keys for authentication
- The archive can be queried to produce task summaries that match conditions, such as task name, monitoring tool used, and resource values comparisons

Resources Archive - Workflows per User

user: gideon

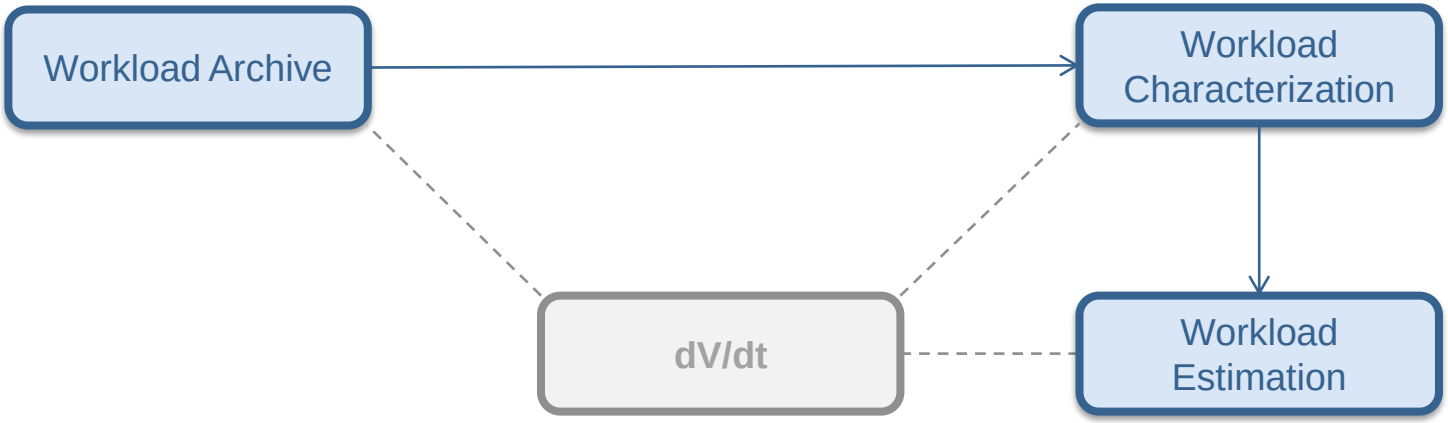
name	hash	type	command
rosetta(76)	22cacabcaf2494a0b70ed4f70016dc93	pegasus	pegasus-plan --conf pegasusrc --dir work --dax dax.xml --sites execution --staging-site CCG --output-site local --cluster horizontal --submit
imputation-mec-pilot-0(72)	3e16fab1377dbcdd4b774d4b63fd52c7	pegasus	pegasus-plan --conf conf/pegasusrc --sites ec2 --dir work/dags --output-site s3 --dax /lfs1/work/page/work/imputation-mec-pilot.dax --nocleanup --input-dir /lfs1/work/page/sample-input --cluster horizontal -vv --force --submit
rosetta(77)	624b453f22c8b6da4fba875bcf90f686	pegasus	pegasus-plan --conf pegasusrc --dir work --dax dax.xml --sites execution --staging-site CCG --output-site local --cluster horizontal --submit

Resources Archive - One Workflow

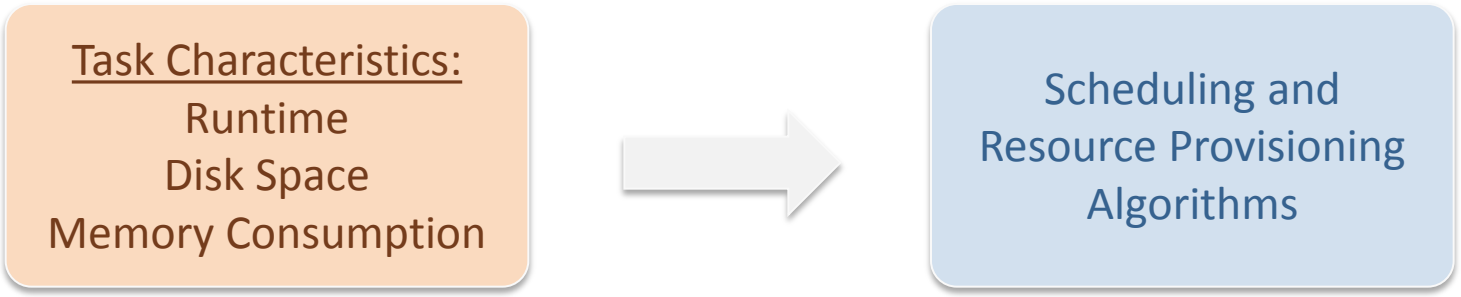
workflow: blast
Found 24 matching entries.
export tasks

command	start	end	wall time (s)	cpu time (s)	concurrent processes	virtual memory (MB)	resident memory (MB)	swap memory (MB)	bytes read	bytes written	files	footprint (MB)
./distributed.script 0	2013-06-28 01:42:34	2013-06-28 02:26:52	2658.065628	2647.76	3	5075	2424	0	5015945881	835584	53	8549
./distributed.script 1	2013-06-28 01:01:54	2013-06-28 02:05:42	3827.227723	3825.77	3	5070	2418	0	10010974054	700416	53	8549
./distributed.script 10	2013-06-27 23:14:24	2013-06-27 23:50:54	2190.215381	2181.61	3	5070	2416	0	10006143297	1155072	53	8549
./distributed.script 11	2013-06-27 22:22:31	2013-06-27 23:01:21	2330.114277	2320.94	3	5078	2425	0	2518945500	380928	53	8549
./distributed.script 12	2013-06-27 22:04:56	2013-06-27 23:16:20	4283.754447	4278.58	3	5090	2413	1	10005309984	380928	53	8549
./distributed.script 13	2013-06-27 23:14:23	2013-06-28 00:32:44	4701.645511	4700.44	3	5075	2424	0	5014224349	454656	53	8549

Workload Modeling and Characterization



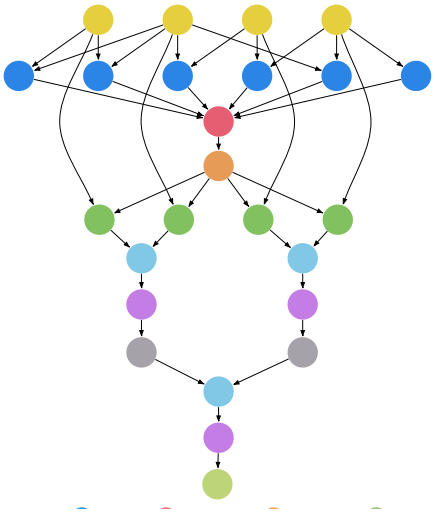
Context



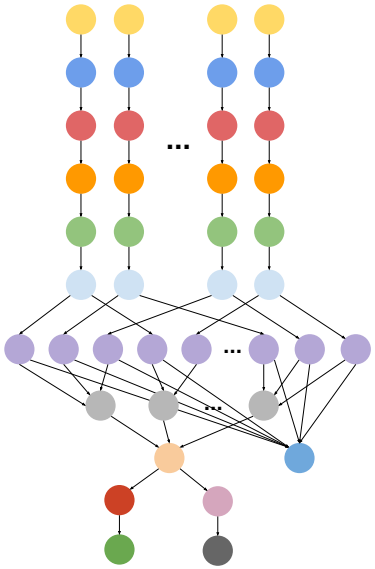
- Methods assume that accurate estimations are available
- A successful workflow execution mainly depends on how tasks are planned and executed
- We propose a method to estimate fine-grained task characteristics online

Scientific Workflows

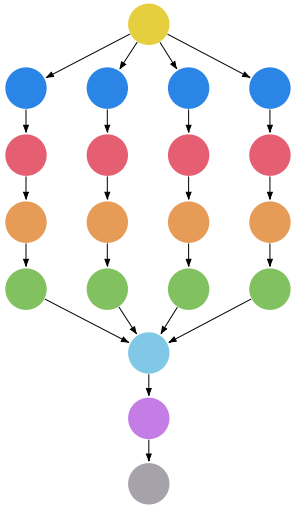
- Directed Acyclic Graph (DAG)
  - Nodes denote tasks
  - Edges denote task dependencies



Montage Workflow



SoyKB Workflow



Epigenomics Workflow

Periodogram Workflow

Rosetta Workflow

Workflow Execution Profiling

Task estimation could be based on mean values

Task	Count	Runtime		I/O Read		I/O Write		Memory Peak	
		Mean (s)	Std. Dev.	Mean (MB)	Std. Dev.	Mean (MB)	Std. Dev.	Mean (MB)	Std. Dev.
mProjectPP	7965	2.59	0.69	4.24	0.19	16.20	0.80	9.96	0.40
mDiffFit	23733	1.25	0.92	24.08	5.76	1.35	1.11	5.32	0.90
mConcatFit	3	122.04	5.27	2.70	0.01	3.15	0.01	7.26	0.01
mBgModel	3	2008.08	88.50	4.14	0.04	0.27	0.00	14.41	0.01
mBackground	7965	2.14	1.68	13.67	6.78	13.05	6.44	11.75	5.78
mImgtbl	51	4.65	2.04	22.64	4.61	0.25	0.05	6.37	0.13
mAdd	51	47.69	14.03	2191.76	560.39	1574.22	383.86	21.66	3.40
mShrink	48	11.53	2.25	835.57	0.31	1.00	0.00	3.05	0.01
mJPEG	3	1.03	0.07	46.18	0.02	0.78	0.00	2.66	0.01

Task estimation based on average may lead to significant estimation errors

Task Estimation Process: Estimate task resource needs based on input data size

Based on Regression Trees

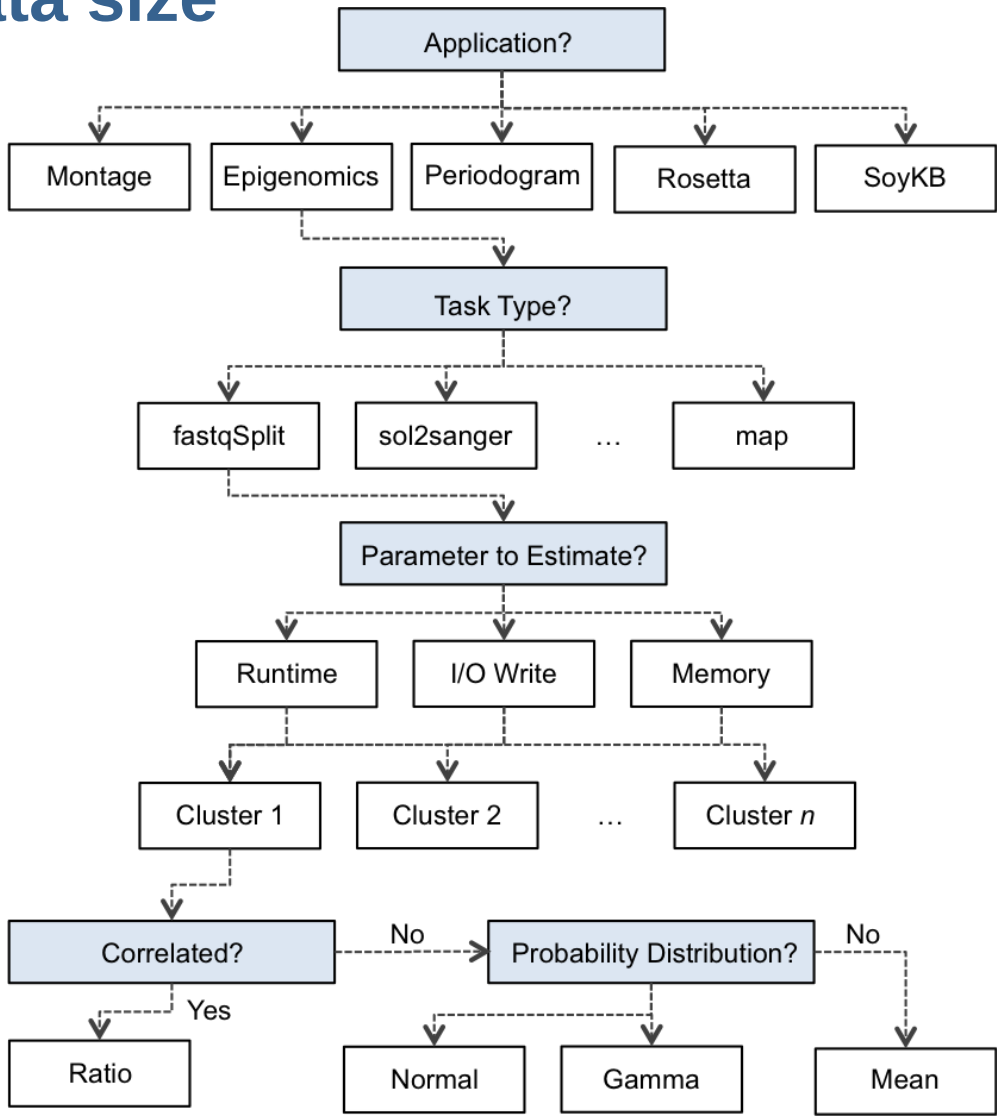
Built offline from historical data analyses

1. If the data is already correlated (e.g., input data and runtime, or input data and output data), no clustering is performed and predictions are done based on the correlation ratio

2. If not, clustering is performed to increase the probability of having subsets where the data is correlated

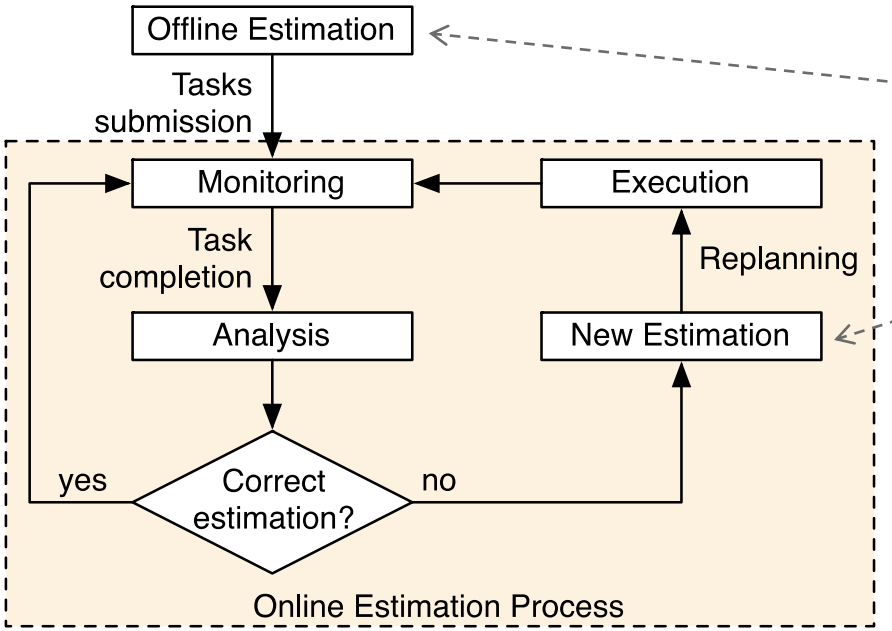
3. If the clustering results in correlated subsets, the ratio is used to perform predictions (as in step 1)

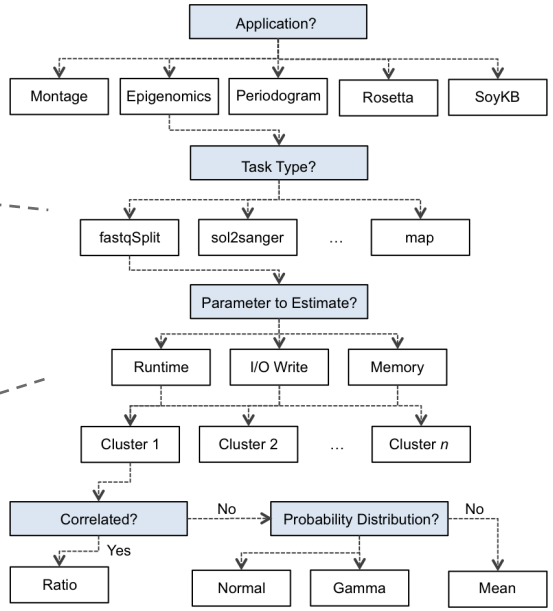
4. If no correlation can be found after clustering, the algorithm tries to identify probability distributions that would describe the subset



Online Estimation Process

- Based on the MAPE-K loop
  - Task executions are constantly monitored
  - Estimated values are updated, and a new prediction is performed





Experiment Conditions

- Trace analysis of 5 workflow applications

- Evaluate the accuracy of our online estimation process
  - **`Offline:`** estimation based on a-priori knowledge
  - **`Online-m`**: estimation based on the median value
  - **`Online-p`**: estimation based on probability distributions
    - Uses the Kolmogorov-Smirnov test (K-S test) to compare empirical data to standard distributions

Experimental Results: SoyKB Workflow

Task	Estimation	Runtime Avg. Error (%)	I/O Write Avg.Error (%)	Memory Avg.Error (%)
alignment_to_reference	Offline	14.73	22.98	10.34
	Online-m	17.31	22.98	10.34
	Online-p	14.73	22.98	10.34
sort_sam	Offline	28.02	19.31	15.50
	Online-m	21.44	4.16	2.65
	Online-p	13.97	4.16	2.65
dedup	Offline	35.11	29.66	21.41
	Online-m	18.76	6.09	5.77
	Online-p	10.01	6.09	5.77
add_replace	Offline	59.55	29.35	25.84
	Online-m	22.14	5.98	4.08
	Online-p	9.08	5.98	4.08
realign_target_creator	Offline	63.22	31.04	40.69
	Online-m	31.18	8.57	10.15
	Online-p	27.83	8.57	10.15
indel_realign	Offline	51.02	20.92	37.41
	Online-m	29.47	3.78	7.09
	Online-p	18.15	3.78	7.09
haplotype_caller	Offline	103.77	94.17	76.23
	Online-m	28.39	7.90	8.44
	Online-p	14.06	7.90	8.44
genotype_gvcfs	Offline	88.50	44.11	51.98
	Online-m	21.96	4.99	5.53
	Online-p	7.14	4.99	5.53
combine_variants	Offline	22.27	30.53	18.34
	Online-m	8.44	5.16	3.10
	Online-p	8.44	5.16	3.10
select_variants_indel	Offline	17.89	16.45	22.32
	Online-m	3.12	9.02	10.43
	Online-p	3.12	9.02	10.43
filtering_indel	Offline	15.70	12.70	10.95
	Online-m	5.86	2.77	3.49
	Online-p	5.86	2.77	3.49
select_variants_snp	Offline	18.01	14.43	24.70
	Online-m	3.03	1.86	10.41
	Online-p	3.03	1.86	10.41
filtering_snp	Offline	13.45	28.14	37.08
	Online-m	2.93	7.29	18.16
	Online-p	2.93	7.29	18.16
merge_gvcf	Offline	37.30	42.68	49.99
	Online-m	4.91	2.04	1.88
	Online-p	4.91	2.04	1.88

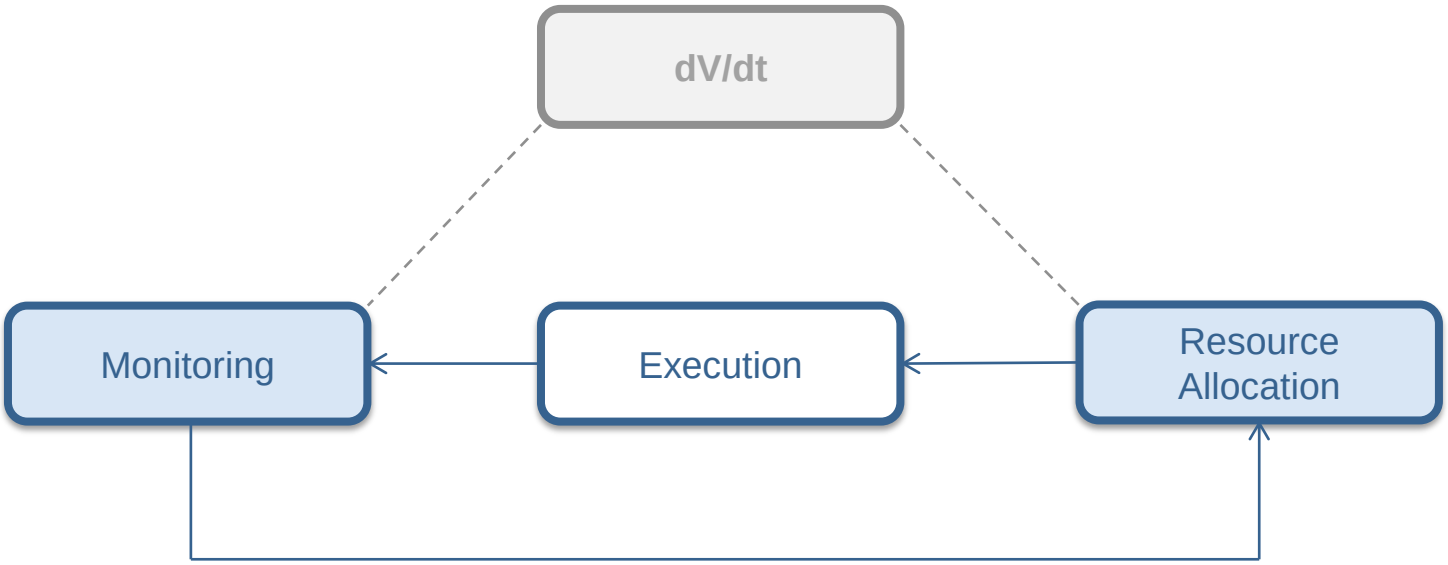
Online Process - Median
Avg. Runtime Error: 20%
Avg. I/O Write Error: 11%
Avg. Memory Error: 14%

Online Process – Probability Distribution
Avg. Runtime Error: 13%
Avg. I/O Write Error: 8%
Avg. Memory Error: 11%

Offline Process
Avg. Runtime Error: 49%
Avg. I/O Write Error: 55%
Avg. Memory Error: 57%

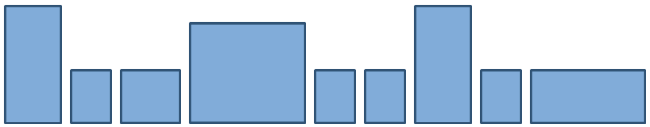
Poor output data estimations leads to a chain of estimation errors in scientific workflows

Provisioning and Resource Allocation



Resource Allocation

- Tasks have different sizes (known at runtime) while computation nodes have fixed sizes



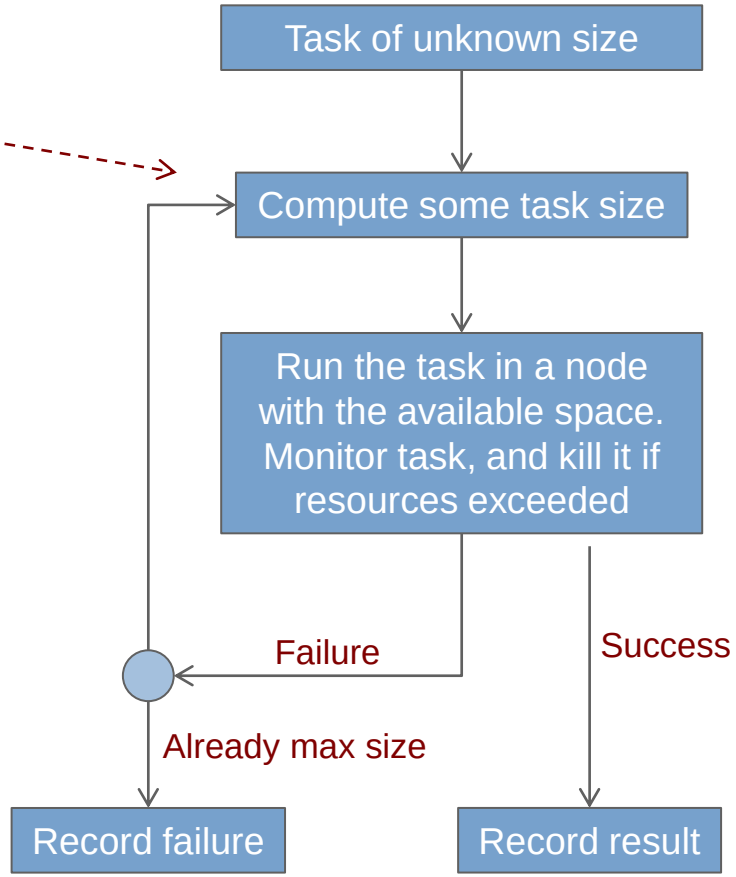
Tasks

Computation Nodes

- Resource allocation strategies
  - One task per node
    - Resources are underutilized
    - Throughput is reduced
  - Many tasks per node
    - Resources are exhausted
    - Jobs fail
    - Throughput is reduced

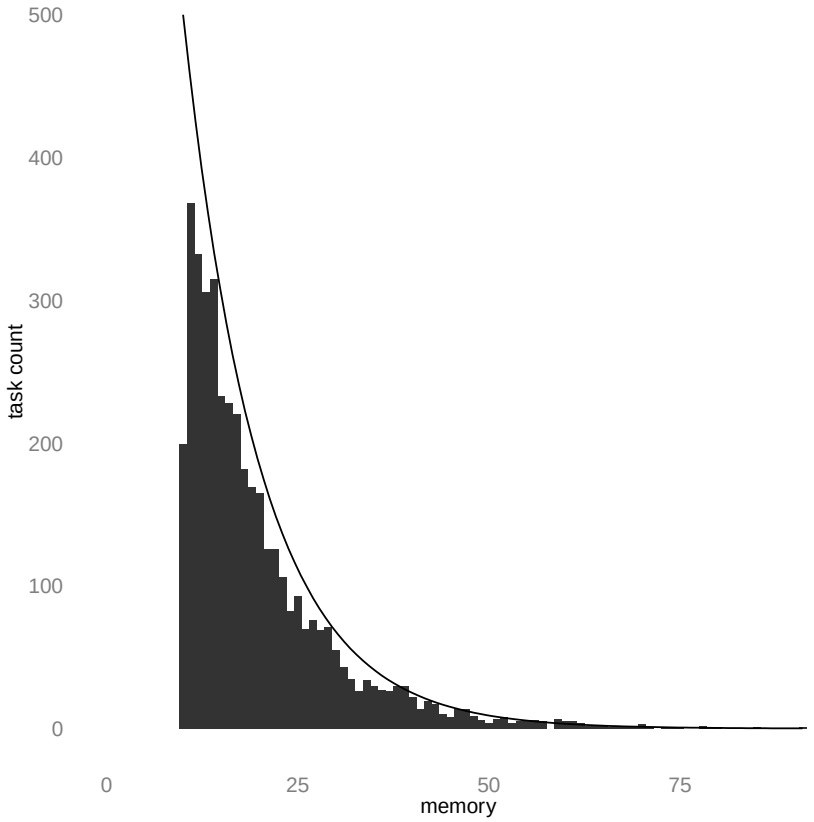
General Approach

- Setting tasks
  - What do we know?
    - Maximum size?
    - Size probability distribution?
    - Empirical distribution?
    - dV/dt Prediction information?
- Our approach
  - Setting task sizes to reduce resource waste
    - Modeling of resource sizes (e.g., memory, disk, or network bandwidth)
    - Assumes the task size distribution is known
    - Adapts to observed behavior

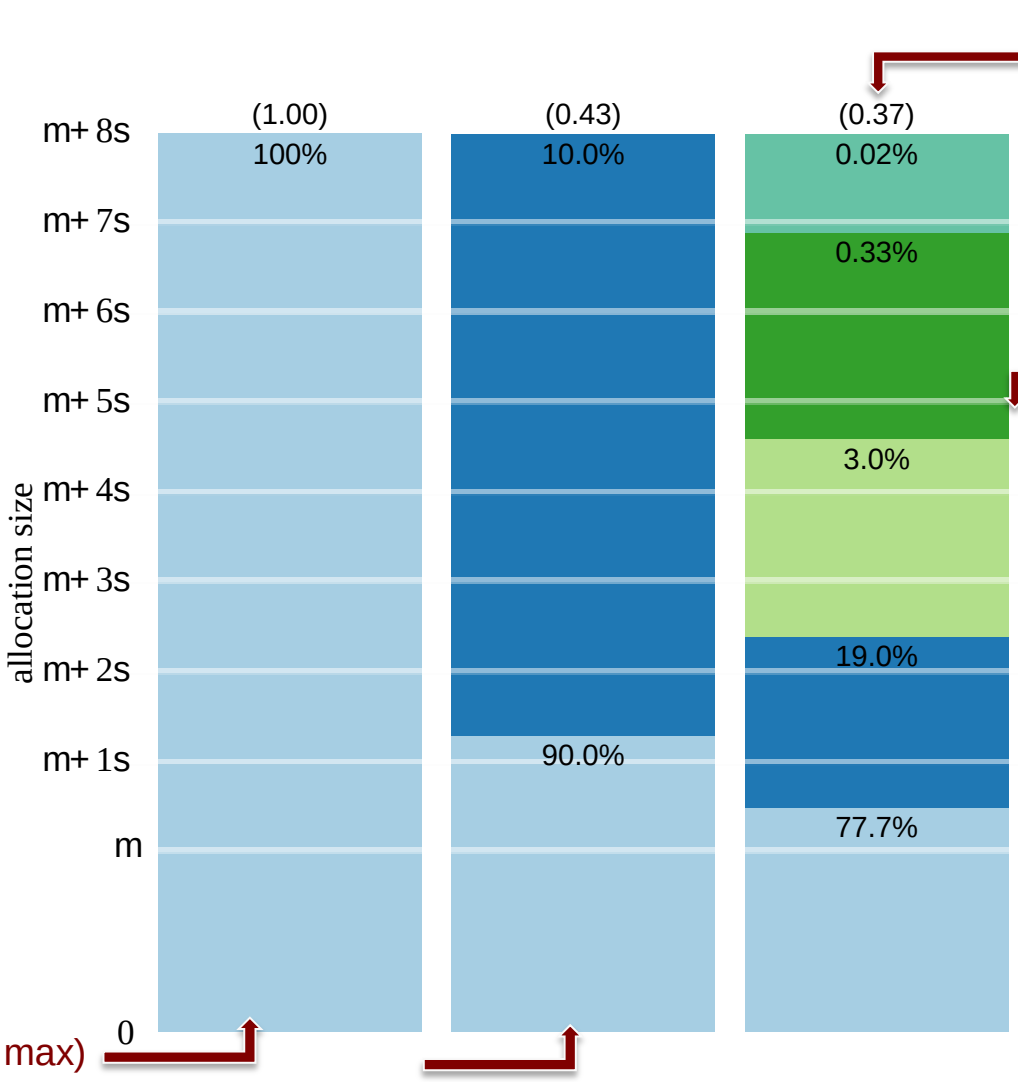


Synthetic Workload Experiment

- Exponential Distribution
  - 5000 Tasks
  - Memory according to an exponential distribution
    - min 10 MB, max 100 MB, average 20 MB
  - Tasks run anywhere from 10 to 20 seconds
  - 100 computation nodes available, from ND Condor pool
  - Each node with 4 cores and a limit of 100 MB of memory



Example: One, Two and Multi-step allocations

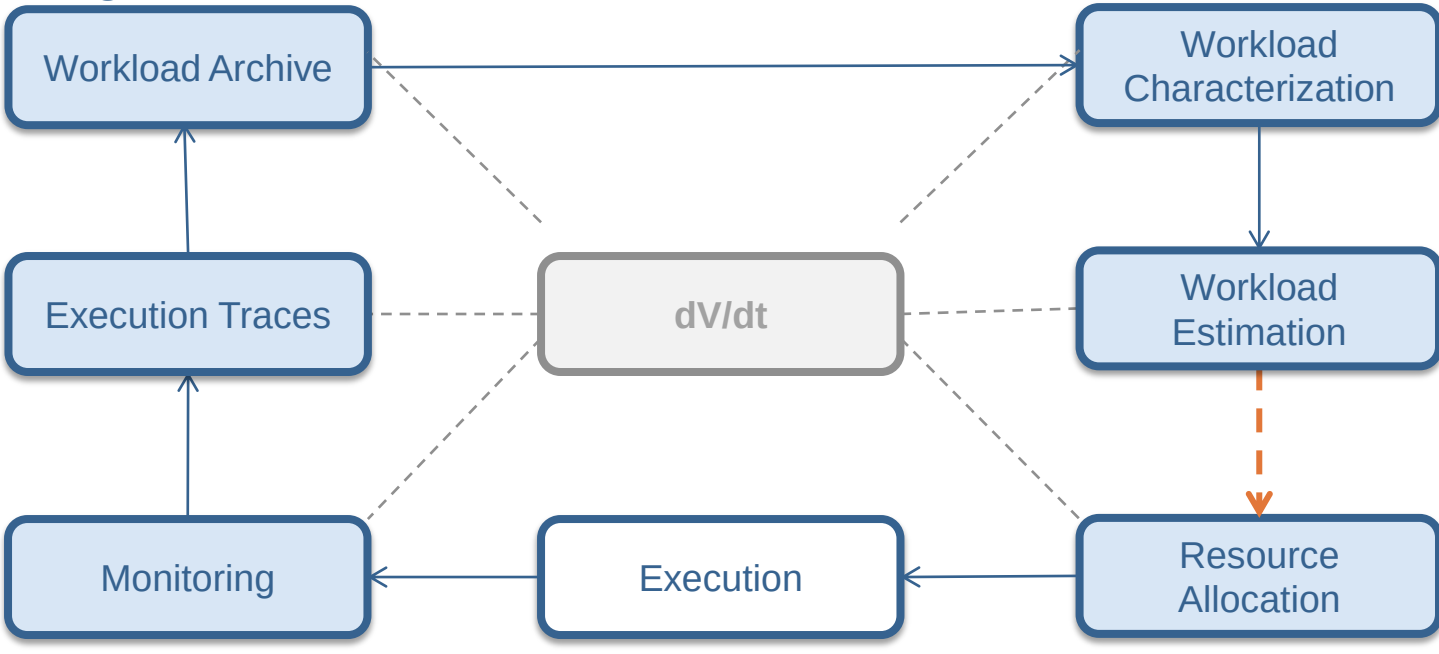


dV/dt Products

- **Monitoring tools:**
  - ***kickstart*** **and** ***resource-monitor*****, support different monitoring methods: ptrace system call interposition, library interposition, polling, support different levels of monitoring information, workflow system independent**
- **Workflow archive:**
  - **Sets of various types workflows with detailed performance information**
  - **Ongoing data collection effort**
- **Methods:**
  - **Online resource need estimation using regression trees and data clustering techniques**
  - **Dynamic resource allocation using runtime behavior information**

Next Steps

- **Enhance monitoring and modeling**
  - Extend modeling to HPC applications
  - Investigate energy consumption
- **Close the loop**
  - Use resource predictions for provisioning and scheduling
  - Improve automation of entire loop
  - Conduct end-to-end experiments with real workloads
- **Productize tools**
  - Turn modeling software into a service



Acknowledgements:

UWM: Miron Livny, Greg Thain

ANL: Bill Allcock

UND: Douglas Thain, Ben Tovar

UCSD: Frank Wuerthwein, James Letts

USC: Ewa Deelman, Gideon Juve, Rafael Ferreira da Silva

https://sites.google.com/site/acceleratingexascale